



State of New Jersey Office of the Attorney General Division of Elections

Design & Configuration Plan for the Statewide Voter Registration System (SVRS)

NEW JERSEY

Deliverable SVRS 024

PRESENTED TO:
MICHAEL GALLAGHER
SVRS PROJECT MANAGER
DEPARTMENT OF LAW AND PUBLIC SAFETY
TRENTON, NEW JERSEY

PRESENTED BY:
COVANSYS CORPORATION
32605 WEST 12 MILE ROAD
FARMINGTON HILLS, MI 48334

MAY 2005

Revision History

Date	Brief Description	Changed By:
04/29/2005	Initial Draft	Anand Balasubramanian
05/02/2005	Reformatted to Template	Wm. Gary Bush
05/12/2005	Final Revisions	Wm. Gary Bush

Table of Contents

Overview	1
Design and Development Processes	3
<i>Collecting the requirements</i>	3
<i>Identifying key design issues</i>	4
<i>Integrating the disciplines</i>	5
Configuration Management	6
<i>Control of changes to requirements, design, and code</i>	6
<i>Control of interface changes</i>	6
<i>Traceability of requirements, design, and code</i>	6
<i>Tools to control versions and builds</i>	7
<i>Parameters for regression testing</i>	7
<i>Baselines established for tools, change logs, and modules</i>	8
<i>Change Request Process</i>	9
<i>Change Control Board and change control process</i>	9
<i>High-Level Timetable of the Process</i>	10

OVERVIEW

*ElectionNet*TM is a “commercial off the shelf” (COTS) product designed to provide state and county election officials with a unified, statewide solution for centralized voter registration and complete election systems management. From voter file management to absentee voting, the system is focused on the security and integrity of the election process. It is designed to automate virtually every aspect of election office operations to maximize productivity, increase efficiency, and standardize election workflow.

The key component of this solution for the New Jersey SVRS project is the Centralized Voter Registration (CVR) module of *ElectionNet*. The *ElectionNet* CVR module was designed to standardize and centralize the registration of voters throughout the State to support online voter registration using a centralized repository, which will decrease voter fraud by eliminating duplicate entry. The system complies with federal statutes of the Help America Vote Act (HAVA) and National Voter Registration Act (NVRA).

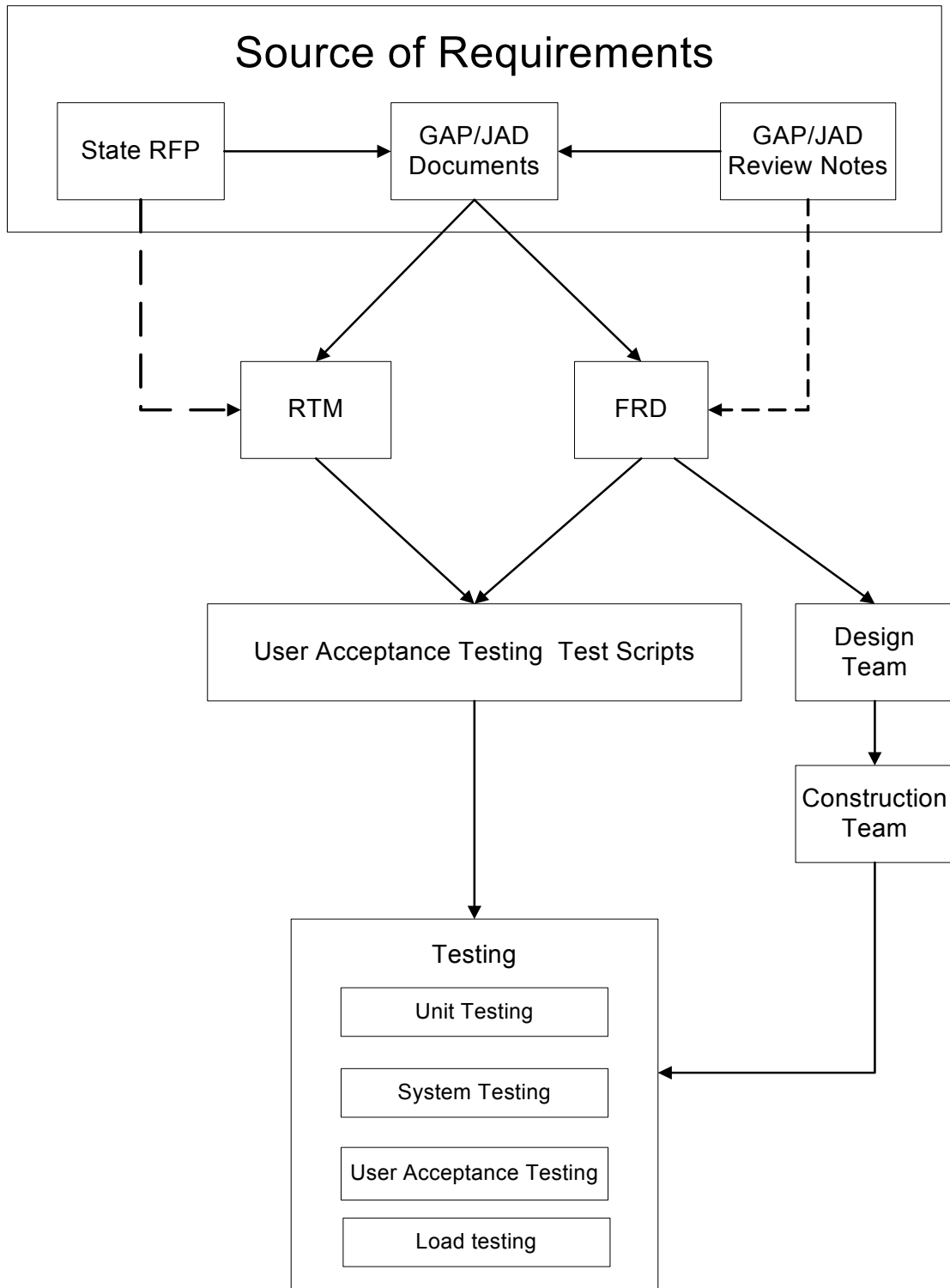
The *ElectionNet* CVR module is built on state-of-the-art technology. The New Jersey SVRS implementation uses IBM WebSphere and Oracle Application Server 9i. It is designed on the principle of providing highly secure, fully featured election functionality through an intuitive user interface. Its user interface provides ease-of-use and flexibility, allowing authorized end users with a username, password, and an Internet connection to quickly process, report, and retrieve information.

ElectionNet's web-services-based architecture is being enhanced to provide a robust county backup solution that can be utilized, if the State server is experiencing an outage or is unavailable due to a communications failure to the county office.

This document provides the overview of the Design and Configuration process to modify the COTS-version of *ElectionNet* to meet the additional and specific requirements for the NJ SVRS.

The collection and definition of the requirements provided within the State's RFQ follows the overall process, as described in the tasks and diagram below:

1. Joint Application Development (JAD) sessions with user representatives to detail needs.
2. Preparation of Gap documents to determine alterations/additions to the COTS product.
3. Summary of all individual requirements with tracking to their source (Requirements Traceability Matrix).
4. Summary of the functional requirements (FRD's) ... i.e., how things work.
5. Review and Approval of the requirements set by the State's core review team.
6. Modification of the COTS application to include all New Jersey requirements.
7. Preparation of test scripts to ensure all functionality and requirements can be exercised in the modified application.
8. Internal testing to verify completeness of the modification and elimination of errors.
9. User Acceptance and Performance Testing by the State's testers to verify conformance to the specifications.



DESIGN AND DEVELOPMENT PROCESSES

Collecting the requirements

During the Business Needs Assessment/Gap Analysis phase, the project team will gather business, functional, and technical requirements through a combination of surveys and requirements sessions (referred to as Joint Application Development (JAD) sessions) with the State's Subject Matter Experts (SMEs), eventual users of the system, and key stakeholders. The Covansys Functional Lead Analyst will guide and facilitate these sessions. Participants will be presented with and will walk through each **Electionet™** screen and function, which comprises the core product, with the sole purpose of gathering and validating State specific requirements. In addition, the Covansys team will solicit information from the participants as to how they think the requirements would best be implemented for New Jersey.

After collecting these requirements, an analysis will be performed to determine the “gaps” that may exist between the **Electionet** application functionality and New Jersey's requirements. A Gap Analysis document will capture the gaps in the application and serve as the basis for application configuration as well as a clearinghouse for gap-related issues. Status of gaps will be maintained, including a measure of priority, impact, relevant details about the gap, and resolutions pertaining to how best to ‘close’ the gap. This information along with the solicited feedback will be critical in completing the analysis and determining the best application design appropriate for New Jersey.

In addition to the Gap Analysis document, Covansys will construct a Requirements Traceability Matrix and Repository to track all requirements specified in the RFP. These requirements will be stored in a Covansys HAVA-specified requirements management database that permits reporting of a specific requirement, selected requirements based on type or attributes, and a complete detailed listing of all requirements. This matrix and the repository will be used throughout the project to assure the design, development, test, and final production system meets the specified requirements.

Identifying key design issues

Once the Gap Analysis document has been reviewed and accepted by the State as representing the required changes, Covansys will develop a conceptual model encompassing detailed designs for the proposed operational solution. Detailed descriptions and modeled representations, where applicable, will be developed and documented to describe how the system will support each identified SVRS functional requirement. These requirements will be presented in a Functional Requirement Document (FRD).

The FRD will be prepared by Covansys Team business analysts working in concert with application developers, and will detail the features and behavior of the target customized application, based on State-specific requirements together with the already resident HAVA requirements. The FRD will include processing logic/flow, business rules, and preliminary screen designs, action buttons, and navigation for those screens requiring changes, as well as including a more complete level of functional detail for areas pertaining to identified gaps. To serve its purpose of driving application changes as well as determining test cases and scenarios that will test every aspect of the application, the FRD will clearly cover the functionality of the entire application. Key design issues identified while conducting these activities will be documented as part of the effort in addition to having been already communicated to the State via the regularly scheduled project status meetings. Working with the State, Covansys will seek reasonable solutions to any design issues to ensure system acceptance and success. By documenting these issues as a function of the FRD, the State will have thorough documentation of what was discussed, what was approved and what was deferred along with reasons should the need arise to revisit any or all decisions. Ultimately, this document will be essential in maximizing the commonality between counties with respect to how the software will be used, which will be critical to the success of the project.

At the same time the FRD is being prepared, Covansys database designers are reviewing the Gap Analysis document and gathered business requirements in order to design the necessary changes to the **ElectioNet** physical database schema to accommodate New Jersey. Entity-Relationship diagrams will be prepared depicting the required changes and containing descriptions of each change. The business rules and logic included in the FRD will map to and will be supported by the required database design. This will address required features currently supported by the **ElectioNet** solution as well as new feature support, with direct traceability back to the requirement documents.

Integrating the disciplines

Integrating the disciplines comes in the form of applying processes and methodologies, which act as the ‘glue’ that melds analysis, design, and development into one cohesive system. The **ElectioNet** product was developed using the Rational Unified Process (RUP) methodology. This methodology establishes and enforces the development of component-based modules. Each **ElectioNet** module is comprised of many independent, scalable, and highly efficient components. As such, it is anticipated to meet the majority of New Jersey’s requirements “out of the box”. For the State-specific requirements, the design specifications (i.e., FRD, Database Design, Interface Design) prepared in the earlier phases are now used to drive application modifications. Business analysts facilitate working sessions with application developers to review these key documents. This ‘technical walkthrough’ provides an opportunity for the developers to ask any questions they may have regarding the requirements in addition to fostering an open and honest exchange of ideas as to how best to meet these requirements.

Each requirement is examined for completeness and preliminary screen designs, depicted in the FRD, are once again reviewed for thoroughness, accuracy, usability, and ability to meet the required functionality. Covansys will communicate to the State Project Sponsor any changes to these screen designs that may be warranted or necessary due to additional information gathered or possible changes to requirements. Once all changes to requirements and/or designs have been approved (via the Change Control process), the application development team will commence with construction/modifications. All development must adhere to strict version control and all code is quality checked via peer and code reviews, documenting recommended changes and providing an audit trail for ensuring quality standards are maintained.

CONFIGURATION MANAGEMENT

Control of changes to requirements, design, and code

Covansys defines Configuration Management to be the discipline of identifying, documenting, reporting and tracking changes that may occur to any aspect of a project (i.e., scope, time or cost) throughout the life cycle of a project. This includes Software Configuration Management (“SCM”), which deals specifically with changes to the software product itself whether from changed requirements, or to support version control or to support changes to environmental factors affecting operational software or networks.

Covansys and the State understand that the spirit of change management is to communicate, document and control the impact of changes to project scope on level of effort, cost, and deadlines. Although some changes may be informal in nature, they still must follow the change control process. Changes to project scope (i.e., requirements, design, code, interfaces, etc.) will be managed through a change control process that will involve investigation, documentation, review and approval, deferral, or rejection of a change request. Please refer to the Consolidated Management Plan under Change Management for a description of the change control procedures.

Any changes to election laws or statutes that are specific to an individual State are handled through the same change control request process. A State, at any time, can request the product to be upgraded due to a change in the State’s election law. Based on the State’s request, a thorough gap analysis will be conducted and based on the findings a formal Change Control Request proposal will be submitted. Based on the State’s approval of the proposal the change to the product will be programmed and delivered according to a mutually agreeable schedule.

Control of interface changes

Controlling and managing change to requirements, design and code to the application proper equally applies to the necessity of similar changes for interfaces. Therefore, Covansys will apply its change management process for interfaces as outlined in the Consolidated Management Plan.

Traceability of requirements, design, and code

As described in section 2 (Design and Development Process), Covansys will construct a Requirements Traceability Matrix and Repository to track all requirements specified in the RFP. These requirements will be stored in a Covansys HAVA-specified requirements management database that permits reporting of a specific requirement, selected requirements based on type or attributes, and a complete detailed listing of all requirements. This matrix and the repository will be used throughout the project to assure the design, development, test, and final production system meets the specified requirements.

Tools to control versions and builds

Covansys recognizes that the ability to control software versions, their builds, migrations and eventual deployment is a critical success factor to the project. Our approach to SCM assists in the establishment and maintenance of the integrity, pertaining to software products. To achieve that goal Covansys and its software partner, PCC Technology, employ the use of Microsoft's Visual SourceSafe for all version and migration control. By providing project-oriented software management, Visual SourceSafe enables the Covansys team to develop applications knowing that their projects and files will be protected. Several features of Visual SourceSafe include:

- Automatically protects and tracks source code, documentation, binaries, and all other file types as they change throughout the software life cycle.
- Full check in and check out features that securely protects files from accidental overwrite by preventing more than one user from modifying the same file at once.
- Versioning features provide snapshots of a project for the quick retrieval of any previous version in the software life cycle.
- Difference reporting provides quick access to changes across separate versions of the same file, enabling developers to know immediately what lines of code have changed.
- Allows team members to reconcile conflicts between different versions of the same file by using a visual merge capability, which provides a point-and-click interface for uniting files and avoids potential loss of valuable changes.

Parameters for regression testing

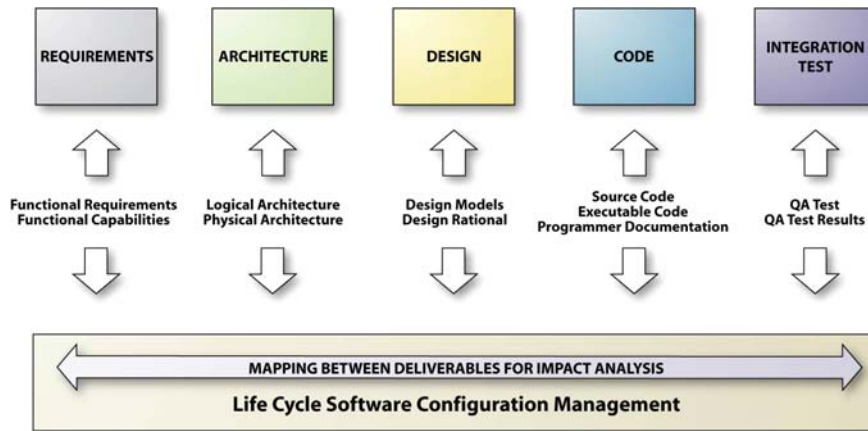
Integration test is a key component of the Project Life Cycle Configuration Management Model. Included in this component are elements necessary to appropriately plan, monitor and control regression testing. Nearly every 'fix or change' requires a certain degree of re-testing. The degree and nature of re-testing is based upon parameters that are a function of the relationship that exists between the components and modules that comprise the software product. The Covansys team has established the following guidelines (i.e., identifying characteristics of critical relationships) for determining the parameters for regression testing:

- Disjoint – Do modules/components exist in a mutually exclusive relationship?
- Hierarchical – Is there a hierarchical structure that demands a top-down or bottom-up cycle of testing?
- Intersection – Does module/component functionality overlap to the degree that common touch points must generate identical results?
- Union – Do individual modules/components function as a single unit in order to complete a single task?
- Independence – Does the occurrence of a particular action in one module/component affect the result (outcome) of actions taken in another module?

Once these questions are answered, parameters are then set regarding testing. Briefly, the answers will dictate the nature and extent of screen navigation paths, action buttons, process flows, business rules and edits (both field-level and application level) to re-test in order to ensure system completeness and integrity.

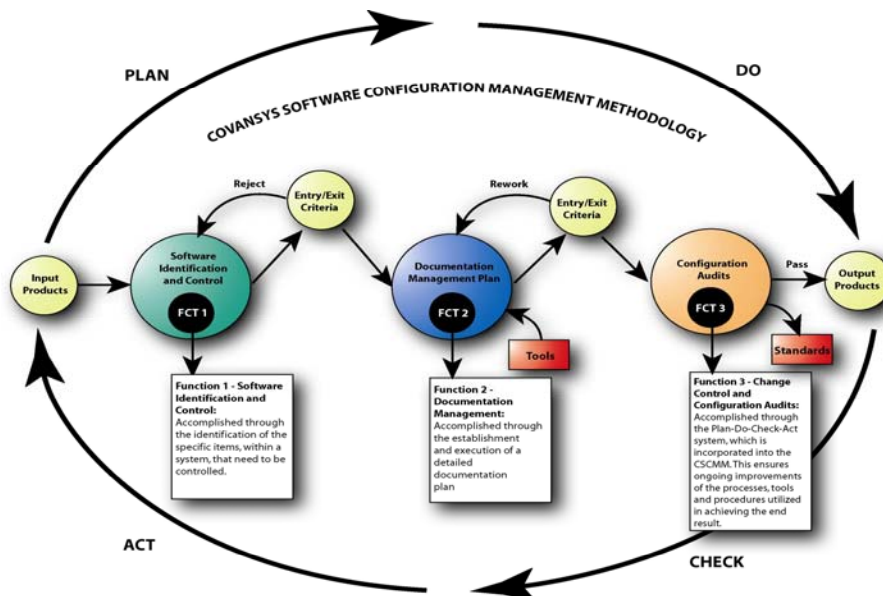
Baselines established for tools, change logs, and modules

At Covansys, we understand Software Configuration Management (SCM) to be the discipline of identifying the configuration of a system at discrete points in time, for systematically controlling changes to this configuration and maintaining the integrity and traceability of this configuration throughout the life cycle of the software system.



Our approach, achieves a one-to-one mapping with the three functions of the Configuration Management Model, in relation to information systems.

Covansys Software Configuration Management Methodology



The approach, based on the Software Engineering Institute (SEI) model, focuses on three areas:

- **Software Identification and Control:** Through this component, an identification scheme is developed by which the structure of the product is reflected. The identification function is addressed through identifying the different and unique application components, giving each a name, a version identifier and a configuration identifier. The control function is achieved through controlling releases of the product throughout the life cycle, by having controls in place that ensure consistent software via the creation of a baseline product.
- **Documentation Management Plan:** Through this component, documents are identified, tracked, traced, versioned, and updated throughout the product lifecycle. This component in particular lends itself to accountability as it deals with the recording and reporting of the status of components, including documents, and change requests. It is also instrumental in gathering vital statistics about the individual modules that comprise the end product. As noted above, Visual SourceSafe includes full documentation features. Visual SourceSafe securely provides an audit trail for every file and every project for all changes. This includes the check-in/check-out process.
- **Configuration Audits:** This aspect validates the completeness of the product and maintains consistency among the various components by ensuring that they exist in a readily accessible and acceptable state throughout the entire project lifecycle. Ultimately, the product is a well-defined collection of these individual components.

Change Request Process

Change control is a very methodical process and, as such, necessitates the need for thorough and accurate documentation and record keeping. Covansys' methodology, as described in the Consolidated Management Plan, provide for the required accountability sought by the State of New Jersey.

Change Control Board and change control process

Please refer to the Consolidated Management Plan for the functions of the Change Control Board and a description of the change control process.

High-Level Timetable of the Process

The project plan for the SVRS provides for the following timetable of the major steps in the Design and Configuration processes for the SVRS:

April 15, 2005 - Completion of JAD sessions (done)

May 20, 2005 - Approval of FRD's (in progress)

May 20, 2005 - Submission of initial RTM (updated throughout the testing process)

June 17, 2005 - Delivery of preliminary test scripts

June 24, 2005 - Completion of preliminary UAT Plan

August 16, 2005 - Completion of COTS modifications and internal testing

August 19, 2005 - Completion of UAT environments and tester training

August 22 - September 12, 2005 - UAT and Performance Test period

September 12 - 16, 2005 - Review and Sign-off of Testing Results

- - -